<u>Amendments to the Claims</u>:

A listing of the entire set of pending claims (including amendments to the claims, if any) is submitted herewith per 37 CFR 1.121. This listing of claims will replace all prior versions, and listings, of claims in the application.

<u>Listing of Claims</u>:

1. - 2. (Cancelled)

3. (Currently Amended) A method for detecting malicious code patterns <u>using a static analysis in consideration of control and data flows,</u> the method comprising: determining whether the values of tokens in two sentences of program code have the same value at the time of execution by one of:

(a) determining<u>, during execution,</u> if both of the tokens <u>of respective macro operations</u> in the two sentences are constants <u>for each block within a control flow graph</u> and if said determination is true, further determining whether relevant token character strings are identical to each other;

(b) determining if one of the tokens <u>of a macro operation</u> in the two sentences is a constant and the other token <u>of a respective macro operation</u> is a variable, and if said determination is true, further determining whether the relevant token character strings are identical to each other after the variable is substituted for the constant <u>by performing a constant propagation and if said determination is true, detecting said malicious code pattern;</u>

(c) determining if both of the tokens in the two sentences are variables and have the same name and range, and if said determination is true, further

-2-

determining whether there are definitions of the relevant variables in a control flow from a preceding one of the two sentences to a following one thereof <u>by performing a copy propagation</u> and if said determination is true, detecting said malicious code pattern;

(d) determining if both of the two tokens of the two sentences are variables but do not have the same name and range, and if said determination is true, further determining whether there are definitions of the relevant variables in a control flow from a preceding one of the two sentences to a following one thereof after the relevant variables are substituted for the original variables <u>by performing a copy propagation</u> and if said further determination is true, detecting said malicious code pattern.

4. (Currently Amended) The method of claim <u>3</u>~~1~~, wherein said determining step (d) of determining whether there are definitions of the relevant variables in a control flow from a preceding one of the two sentences to a following one thereof after the relevant variables are substituted for the original variables, further comprises performing a copy propagation to substitute relevant variables for original variables.

5. (Currently Amended) The method of claim <u>4</u>~~2~~, wherein said copy propagation reduces the number of copies by finding a variable which will always have a specific constant value upon execution of a program and performing substitution through a copy sentence in the form of "x=y".

6.   (Previously Presented) The method of claim 5, wherein the copy propagation is performed via a data flow analysis in a created control graph to create a modified control graph.

7.   (Currently Amended) The method of claim 3~~1~~, wherein said determining step (b) of determining whether there are definitions of the relevant variables in a control flow from a preceding one of the two sentences to a following one thereof further comprises performing a constant propagation to substitute relevant variables for original variables.

8.   (Previously Presented) The method of claim 7, wherein said constant propagation finds a variable or formula that will always have a specific constant value upon execution of a program.

9.   (Previously Presented) The method of claim 7, wherein the constant propagation is performed via a data flow analysis in a created control graph to create a modified control graph.

10. (New) A method for detecting malicious code patterns using a static analysis in consideration of control and data flows, the method comprising: determining whether the values of tokens in two sentences of program code

have the same value at the time of execution by one of:

(a) determining, during execution, if both of the tokens of respective macro operations in the two sentences are constants for each block within a control flow graph and if said determination is true, further determining whether relevant token character strings are identical to each other;

(b) determining if one of the tokens of a macro operation in the two sentences is a constant and the other token of a respective macro operation is a variable, and if said determination is true, further determining whether the relevant token character strings are identical to each other after the variable is substituted for the constant by performing a constant propagation and if said determination is true, detecting said malicious code pattern;

(c) determining if both of the tokens in the two sentences are variables and have the same name and range, and if said determination is true, further determining whether there are definitions of the relevant variables in a control flow from a preceding one of the two sentences to a following one thereof by performing a copy propagation and if said determination is true, detecting said malicious code pattern;

(d) determining if both of the two tokens of the two sentences are variables but do not have the same name and range, and if said determination is true, further determining whether there are definitions of the relevant variables in a control flow from a preceding one of the two sentences to a following one thereof after the relevant variables are substituted for the original variables by performing a copy propagation to substitute relevant variables for original variables and if said further determination is true, detecting said malicious code pattern.

11.   (New) The method of claim 10, wherein said copy propagation reduces the number of copies by finding a variable which will always have a specific constant value upon execution of a program and performing substitution through a copy sentence in the form of "x=y".

12. (New) The method of claim 10, wherein the copy propagation is performed via a data flow analysis in a created control graph to create a modified control graph.

13.(New) The method of claim 10, wherein said determining step (b) of determining whether there are definitions of the relevant variables in a control flow from a preceding one of the two sentences to a following one thereof further comprises performing a constant propagation to substitute relevant variables for original variables.

14.(New) The method of claim 13, wherein said constant propagation finds a variable or formula that will always have a specific constant value upon execution of a program.

15.(New) The method of claim 13, wherein the constant propagation is performed via a data flow analysis in a created control graph to create a modified control graph.